CRL at NTCIR2

Masaki Murata, Masao Utiyama, Qing Ma, Hiromi Ozaku, Hitoshi Isahara
Communications Research Laboratory
2-2-2 Hikaridai Seika-cho, Soraku-gun,
Kyoto, 619-0289 Japan
{murata,mutiyama,qma,romi,isahara}@crl.go.jp

Abstract

We have developed systems of two types for NTCIR2. One is an enhenced version of the system we developed for NTCIR1 and IREX. It submitted retrieval results for JJ and CC tasks. A variety of parameters were tried with the system. It used such characteristics of newspapers as locational information in the CC tasks. The system got good results for both of the tasks. The other system is a portable system which avoids free parameters as much as possible. The system submitted retrieval results for JJ, JE, EE, EJ, and CC tasks. The system automatically determined the number of top documents and the weight of the original query used in automatic-feedback retrieval. It also determined relevant terms quite robustly. For EJ and JE tasks, it used document expansion to augment the initial queries. It achieved good results, except on the CC tasks.

Keywords: *newspaper article, locational information, portable system, flexible system,*

1 Introduction

We have developed two systems for the second NTCIR Workshop's information retrieval (IR) tasks.

One is an enhanced version of the system that was used for the first NTCIR Workshop's IR tasks [5] and the IREX Workshop's IR tasks [6]. We call this System A. The other is a newly developed system in which free parameters are avoided as much as possible. We call this System B.[1]

System A participated in tasks set in Japanese and Chinese (JJ and CC). It achieved high average precisions on both tasks. System B participated in tasks set in Japanese, English, and Chinese (JJ, JE, EE, EJ, and CC). It achieved high average precisions on the JJ, EE, JE, and EJ tasks.

Although the two systems participated in some of the same tasks, the details of the system implementations are rather different. Thus, we describe the two systems separately, focusing on particular tasks; i.e., we describe System-A in the context of CC tasks and describe System-B in the context of JJ, EE, JE, and EJ tasks.

2 Chinese IR Tasks

In this section, we describe System A in the context of CC tasks. System A participated in JJ tasks[2] and CC tasks, and achieved particularly good results on the CC tasks. This reason is that the types of documents used in the CC tasks were very different from those used in the JJ tasks. While the JJ tasks involved retrieval from a database of academic conference papers, the CC tasks involved retrieval from a database of newspaper articles. System A[3] takes advantage of such characteristics of newspapers as the title or the first sentence of the body of an article in a newspaper often indicating the article's subject. We thus expected System A to be effective on the CC tasks. In the following sections, we give a detailed description of System A and report on the experimental results of System A's application to the CC tasks.

2.1 Outline of System A

System A uses Robertson's 2-poisson model [9] which is one kind of probabilistic approach. In Robertson's method, each document's score is calculated by using the following equation.[4] The documents that

[1] System A was developed mainly by the first author, and System B was developed mainly by the second author.

[2] System A particpated in the long-query and short-query JJ tasks. The best average precisions of the two tasks in terms of A judgement were 0.4082 (CRL20) and 0.3730 (CRL16), and the best average R-precisions were 0.4210 (CRL20) and 0.3866 (CRL27). These results are also good. Strings in parentheses indicate system ids in the NTCIR contest. Examination of System A's performance on JJ tasks is the subject of a forthcoming publication.

[3] System A is based on the system we entered in the IREX contest. In the IREX contest, articles in a database of newspapers database were used as the test collection. System A achieved good results in the IREX contest, too [6, 7].

[4] This equation is BM11, which corresponds to BM25 in the case of $b = 1$ [11].

obtain high scores are then output as retrieval results. ($Score(d,q)$ below is the score of a document d against a query q.)

$$Score(d,q) = \sum_{\substack{\text{term } t \\ \text{in } q}} \left(\frac{tf(d,t)}{tf(d,t) + k_t \frac{length(d)}{\Delta}} \times log \frac{N}{df(t)} \times \frac{tf_q(q,t)}{tf_q(q,t) + kq} \right) \quad (1)$$

where t indicates a term that appears in a query. $tf(d,t)$ is the frequency of t in a document d, $tf_q(q,t)$ is the frequency of t in a query q, $df(t)$ is the number of the documents in which t appears, N is the total number of documents, $length(d)$ is the length of a document d, and Δ is the average length of the documents. k_t and k_q are constants which are set according to the results of experiments.

In this equation, we call $\frac{tf(d,t)}{tf(d,t) + k_t \frac{length(d)}{\Delta}}$ the TF term, (abbr. $TF(d,t)$), $log \frac{N}{df(t)}$ the IDF term, (abbr. $IDF(t)$), and $\frac{tf_q(q,t)}{tf_q(q,t)+kq}$ the TF$_q$ term (abbr. $TF_q(q,t)$).

In System A, several terms are added to extend this equation, and its method is expressed by the following equation.

$$Score(d.q) = K_{category}(d) \left\{ \sum_{\substack{\text{term } t \\ \text{in } q}} (TF(d,t) \times IDF(t) \times TF_q(q,t) \times K_{location}(d,t) \times \left(log \frac{Nq}{qf(t)} \right)^{k_{Nq}} \right) + \frac{length(d)}{length(d) + \Delta} \right\} \quad (2)$$

The TF, IDF and TF$_q$ terms in this equation are identical to those in Eq. (1). The value of the term $\frac{length}{length+\Delta}$ increases with the length of the document. This term is introduced because if all of the other information is exactly the same, the longer document is more likely to include content that is a relevant response to the query. Nq is the total number of queries and $qf(t)$ is the number of queries in which t occurs. Those terms which occur more frequently in queries are more likely to be stop words such as "documents" and "thing." We decrease the scores of stop words by using $log \frac{Nq}{qf(t)}$. $K_{category}$ and $K_{location}$ are extended numerical terms that are introduced to improve precision of results. $K_{category}$ uses the category information of the document found in newspapers, such as the economic or political pages. $K_{location}$ uses the location of the term within the document. If the term is in the title or at the beginning of the body of the document, it is given a higher weighting. In the next section, we explain these extended numerical terms in detail.

2.2 Extended numerical terms

We use the two extended numerical terms $K_{location}$ and $K_{category}$ as shown in Eq. (2). In this section, they are explained in detail.

1. Location information ($K_{location}$)

   In general, the title or the first sentence of the body of a document in a newspaper indicates its subject. Therefore, the precision of information retrieval can be improved by assigning more weight to the terms from these two locations. This is achieved by $K_{location}$ which adjusts the weight on a term the basis of whether or not it appears at the beginning of the document. If a term is in the title or at the beginning of the body, it is given a high weighting. Otherwise, it is given a low weighting. $K_{location}$ is expressed as follows:

$$K_{location}(d,t) = \begin{cases} k_{location,1} \\ \text{(when a term } t \text{ occurs in the title of a document } d\text{)}, \\ 1 + k_{location,2} \frac{(length(d) - 2 * P(d,t))}{length(d)} \\ \text{(otherwise)} \end{cases} \quad (3)$$

   $P(d,t)$ is the location of a term t in the document d. When a term appears more than once in a document, its first appearence is used. $k_{location,1}$ and $k_{location,2}$ are constants which are set according to the results of experiments.

2. Categorical information ($K_{category}$)

   $K_{category}$ uses category information such as whether or not the document appears on the economic or political pages. This operates by applying the technique called relevance feedback [13]. Firstly, we specify the categories which occur in the top 15 documents of the first retrieval when $K_{category} = 1$. Then, we increase the scores of documents that are in majority or most-frequent categories. For example, the top 15 documents of the first retrieval were most often from the economic pages, we increase the scores of a documents from economic pages and decrease the scores of all documents from other sections of the newspaper. $K_{category}$ is expressed as follows;

$$K_{category}(d) = 1 + k_{category} \frac{(RatioA(d) - RatioB(d))}{(RatioA(d) + RatioB(d))} \quad (4)$$

   where $RatioA$ is the proportion of the top 100 documents in a given category on the first retrieval. $RatioB$ is the proportion of that category

in all the documents. The value of $K_{category}(d)$ is large when $RatioA$ is large (the top 100 documents of the first retrieval frequently appear on the same pages as a document d.) and $RatioB$ is small (few of the documents appear on the same pages as d). $k_{category}$ is a constant which is set according to the results of experiments.

2.3 How terms are extracted

Before being able to use Eq. (2) in information retrieval, we must extract terms from a query. This section describes how this is done. With regard to term extraction, we considered the several methods listed below.

1. Method of using only the shortest terms

   This is the simplest method. In the method, the query sentence is divided into short terms by using a morphological analyzer or a similar tool. All of the short terms are used in the retrieval process. The method used to divide the query sentence into short terms is described in Section 2.4.

2. Method of using all term patterns

   In the first method the terms are too short. For example, "enterprise" and "amalgamation" would be used instead of "enterprise amalgamation."[5] We felt that "enterprise amalgamation" should be used along with the two short terms. Therefore, we decided to use both short and long terms. We call this the "all term-patterns method." For example, when "enterprise amalgamation materialization" was input, we used "enterprise", "amalgamation", "materialization", "enterprise amalgamation", "amalgamation materialization", and "enterprise amalgamation materialization" as terms for information retrieval. We felt that this method would be effective because it makes use of all term patterns. We also felt, however, that it is inequitable that only the three terms "enterprise," "amalgamation," "materialization," are derived from "... enterprise ... amalgamation ... materialization ...", while six terms are derived from "enterprise amalgamation materialization." We examined several methods of normalization in preliminary experiments, then decided to divide the weight of each term by $\sqrt{\frac{n(n+1)}{2}}$, where n is the number of successive words. For example, in the case of "enterprise amalgamation materialization", $n = 3$.

[5] Although this part of the paper deals only with retrieval from Chinese-language texts, and not English, we have used English examples for the benefit of this English-lanugae journal's readers. This method handles compound nouns and can be applied not only to Chinese but also to English.

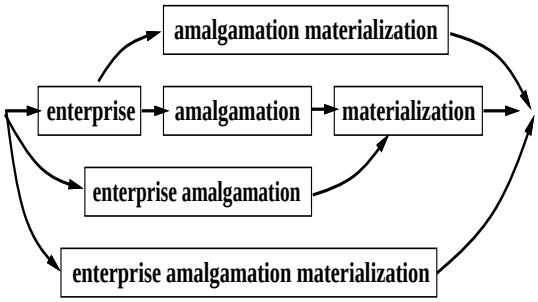

Figure 1. An example of a lattice structure

3. Method using a lattice

   Although the method of using all-term patterns effectively uses all patterns of terms, it needs to be normalized by using the adhoc equation $\sqrt{\frac{n(n+1)}{2}}$. We thus considered a method in which all term patterns are stored in a lattice. We used the patterns in the path with the highest score on Eq. (2). (This method is almost the same as Ozawa's [8]. The differences are the fundamental equation for information retrieval, and whether or not a morphological analyzer is used.)

   For example, in the case of "enterprise amalgamation materialization" the lattice shown in Fig. 1 is obtained. As shown in this figure, the score is calculated for each of the four paths by using Eq. (2), and the terms in the highest-scoring path are used. This method does not require the adhoc normalization required by the method of using all term patterns.

4. Method of using down-weighting [3]

   This is the method that Fujita proposed at the IREX contest [14]. It is similar to the all-term patterns method. It uses all term patterns but the method of normalization is different from that used in the all-term patterns method. The weights of the shortest terms are kept constant while the weights of the longer terms are decreased. We decided to apply the weight ${k_{down}}^{x-1}$ to such terms, where x is the number of shortest terms and k_{down} was set according to the results of experiments.

2.4 The method dividing the query sentence into short terms

We used the following three methods to divide the query sentence into short terms.[6]

[6] System A only segments sentences of documents are not segmented except for automatic feedback.

1. Using a morphological analyzer

   In this method, the query sentence is segmented by using the CSeg&Tag 1.0 Chinese-language morphological analyzer [17].

2. Segmentation by using mutual information

   This method is based on the method [16] proposed by Sproat et al. It calculates the mutual information of two adjacent characters and divides them when their mutual information. The details of our method are as follows.

   Almost all Chinese words consist of one Chinese character or two Chinese characters.[7] So we assumed that all terms consist of one Chinese character or two Chinese characters. Thus, our method firstly divides Chinese sentences into fragments which consist of one Chinese character or two Chinese characters by using mutual information. This is done by repeatly applying the following procedure.

   - Divide up pairs of adjacent characters with the lowest amount of mutual information, where each pair is part of a fragment which consist of more than two Chinese character.

   Next, we use the statistics of the Chinese corpus. In this case, we assume that the ratio of one-character words and two-characters words in a Chinese text is a:b.[8] We take this statistic then re-divide those fragments that consist of pairs of characters having little mutual information into two separate one-character words in such a way that our process of division produces a text broken up into one- and two-character words in the approximate proportion a:b. This is done by repeating the following procedure until the text will be divided up to produce the approximate proportion a:b.

   - Divide those fragments consisting of pairs of characters having the lowest mutual information

   The result of this procedure is equivalent to that of the following procedure.

   - Divide up those fragments consisting of pairs of characters having a level of mutual information which is equal to or lower than k_{cmi}, where k_{cmi} is the amount of mutual information that will divide up the text to produce the approximate proportion a:b.

3. Using both of the above two methods

   This method firstly divides up the Chinese sentences by using the morphological analyzer and then further divides up the fragments by using mutual information and the statistics on the Chinese corpus.

[7] According to the paper [16], the occurrence rate of words which consist of three Chinese characters is under 1%.

[8] For example, Spraot stated that this ratio is about 7:3 [16].

2.5 Automatic feedback in System A

Automatic feedback is also used in System A. In System A, an element of automatic feedback is inclued via the IDF term of the equation (2). When performing automatic feedback, we substitute the following equation for the original IDF term.

$$IDF(t) = \{E(t) + k_{af} \times (Ratio\ C(t) - Ratio\ D(t))\} \times IDF_{orig}(t) \quad (5)$$

$$E(t) = \begin{array}{l} 1 \text{ (when a term t is in a query)} \\ 0 \text{ (otherwise)} \end{array} \quad (6)$$

where $Ratio\ C(t)$ is the proportion of the top k_r documents of the first retrieval in which a term t appears. $Ratio\ D(t)$ is the proportion of all of the documents in which a term t appears. $IDF_{orig}(t)$ is the original IDF term. This formula is based on Rocchio's formula [12]. k_{af} and k_r are constants set according to the results of experiments.

Term expansion is also used in System A. The terms 'Terms' as defined below are added.

$$Terms = \{t | P(t) \geq k_p\} \quad (7)$$

where $P(t)$ is the probability that a term t appears in no less than n documents of the top k_r documents. $P(t)$ is approximately calculated by assuming that the appearance of the term t follows a binominal distribution with a probability of the occurrence rate of the term t in all the documents. k_p is a constant set according to the results of experiments.

2.6 Weighting counting in automatic feedback

We considered that a term which occurs in a document which has a higher rank on the first retrieval is more important. So, when counting the frequency of a term t in a document d with a rank of $Rank(d)$, System A applied the following factor $AFW(t,d)$ to the frequency.

$$AFW(t,d) = (k_{afw} + 1) - 2 \times k_{afw} \frac{Rank(d) - 1}{k_r - 1} \quad (8)$$

where k_{afw} is a constant set according to the results of experiments. Equations (5) and (7) are calculated by using the frequency calculated by Equation 8.

2.7 Experiments

The experimental results of System A are shown in Table 1. "LO", "SO", "VS", and "TI" indicate a long-query task, a short-query task, a very short query

Table 1. Experimental results in CC Tasks

			parameters								R-Precsision		Ave. Presision	
	Task	ID	Term	k_{mi}	k_{Nq}	dw	af	L	C	k_r	k_{af}	rigid	relax	rigid	relax
S1	LO	07	MI	4.5	0	y	y	y	y	5	0.7	0.5751	0.6630	0.6348	0.7261
S2	LO	–	MI	4.5	0	y	n	y	y	5	0.7	0.5529	0.6564	0.6186	0.7146
S3	LO	–	MI	4.5	0	n	n	y	y	5	0.7	0.5660	0.6572	0.6183	0.7118
S4	LO	08	MI	4.5	1	y	y	y	y	5	0.7	0.5842	0.6692	0.6392	0.7362
S5	LO	09	MI	4.5	t	y	y	y	y	5	0.7	0.5803	0.6651	0.6386	0.7342
S6	LO	02	MI	3	0	y	y	y	y	5	0.7	0.5812	0.6685	0.6439	0.7326
S7	LO	03	MI	3	0	y	n	y	y	5	0.7	0.5632	0.6699	0.6329	0.7231
S8	LO	04	MI	3	0	n	y	y	y	5	0.7	0.5865	0.6684	0.6438	0.7325
S9	LO	05	MI	3	0	n	n	y	y	5	0.7	0.5587	0.6695	0.6329	0.7229
S10	LO	06	MI	3	1	y	y	y	y	5	0.7	0.5782	0.6813	0.6459	0.7409
S11	LO	10	MI	3	t	y	y	y	y	5	0.7	0.5780	0.6724	0.6427	0.7383
S12	LO	19	MI	4	1	y	y	y	y	5	0.7	0.5814	0.6767	0.6407	0.7399
S13	LO	–	MI	4	1	y	y	n	y	5	0.7	0.5659	0.6704	0.6316	0.7334
S14	LO	–	MI	4	1	y	y	y	n	5	0.7	0.5916	**0.6945**	**0.6567**	**0.7488**
S15	LO	–	MI	4	1	y	y	n	n	5	0.7	0.5778	0.6822	0.6530	0.7445
S16	LO	18	MI	4	1	y	y	y	y	5	1	0.5900	0.6752	0.6415	0.7387
S17	LO	20	MI	4	1	y	y	y	y	7	0.7	0.5746	0.6778	0.6388	0.7374
S18	LO	21	MI	4	1	y	y	y	y	10	0.7	0.5605	0.6741	0.6299	0.7316
S19	LO	11	MI	4	1	y	y	y	y	15	0.7	0.5743	0.6776	0.6265	0.7291
S20	LO	12	MI	4	1	y	y	y	y	20	0.7	0.5577	0.6767	0.6254	0.7268
S21	LO	13	MI	4	1	y	y	s	s	5	0.7	0.5709	0.6703	0.6203	0.7271
S22	LO	14	T+M	4	1	y	y	y	y	5	0.7	0.5924	0.6810	0.6486	0.7413
S23	LO	–	TAG	4	1	y	y	y	y	5	0.7	**0.5936**	0.6803	0.6501	0.7419
S24	LO	15	T+M	4	1	y	n	y	y	5	0.7	0.5820	0.6778	0.6388	0.7290
S25	LO	17	T+M	4	1	n	y	y	y	5	0.7	0.5712	0.6739	0.6341	0.7276
S26	LO	16	T+M	4	1	n	n	y	y	5	0.7	0.5557	0.6628	0.6165	0.7145
S27	SO	02	MI	4	1	y	y	y	y	5	0.7	0.5831	**0.6817**	0.6340	0.7368
S28	SO	03	T+M	4	1	y	y	y	y	5	0.7	**0.5974**	0.6766	**0.6529**	**0.7376**
S29	VS	02	MI	4	1	y	y	y	y	5	0.7	0.5990	**0.6788**	0.6516	0.7387
S30	VS	03	T+M	4	1	y	y	y	y	5	0.7	**0.6089**	0.6749	0.6596	0.7397
S31	VS	–	T+M	4	1	y	y	n	y	5	0.7	0.5893	0.6669	0.6468	0.7282
S32	VS	–	T+M	4	1	y	y	y	n	5	0.7	0.6027	0.6781	**0.6722**	**0.7454**
S33	VS	–	T+M	4	1	y	y	n	n	5	0.7	0.5889	0.6636	0.6563	0.7350
S34	VS	–	TAG	4	1	y	y	y	y	5	0.7	0.6086	0.6757	0.6604	0.7399
S35	TI	02	MI	4	1	y	y	y	y	5	0.7	**0.4683**	**0.5923**	**0.4813**	**0.6239**
S36	TI	03	T+M	4	1	y	y	y	y	5	0.7	0.4651	0.5770	0.4793	0.6118

The number of queries is 50. The number of documents is 132,173.

task, and a title-query task. The column "ID" indicates the system id in the NTCIR 2 contest. "–" in "ID" indicates a system which was not submitted for the formal run of the NTCIR 2 contest. The column "Term" indicates the method used to divide the query sentence up into short terms. "TAG", "MI", and "T+M" respectively indicate the use of the Chinese morphological analyzer, mutual information, and both the morphological analyzer and mutual information. k_{cmi},[9] k_{Nq}, k_r, and k_{af} are set as in Table 1. "dw", "af", "L" and "C" indicate the down-weighting method, automatic feedback method, locational information, and categorical information. "y" in a column indicates the use of the method, and "n" indicates that the method was not used. When we do not use the down-weighting method, we use the shortest-terms method as the method of extracting terms.[10] The other parameters are set as follows: $k_{location,1} = 1.2$, $k_{location,2} = 0.1$, $k_{category} = 0.1$, $k_t = 1$, $k_q = \infty$, $k_p = 0.9$, and $k_{afw} = 0.5$. "s" in "L" and "C" means the strong setting where $k_{location,1} = 1.3$, $k_{location,2} = 0.15$, $k_{category} = 0.15$. "t" in "k_{Nq}" means using $log\frac{Nq}{qf(t)}$ in a more complex way such that $qf(t)$ means the number of queries whose titles contain a term t.

The following were the findings produced by the experimental results.

[9] In the CHIR newspapers database, using k_{cmi} = 5.33, 4.96, 4.56, 4.10, and 3.53 divides up the text to produce the approximate proportions of 7:3, 6.5:3.5, 6:4, 5.5:4.5, and 5:5.

[10] Our previous work [7] had confirmed that the use of all term patterns is not a good method, and that even the simple method of using only the shortest terms can achieve good results.

- The precisions of "T+M" or "TAG" are slightly higher than that of "MI." We thus found that using the morphological analyzer produced better results than using mutual information.
- By comparing S12 with S13 or S30 with S31, we found that locational information achieved an improvement of about 0.02 or 0.03. We can see that locational information is very effective.
- By comparing S12 with S14 or S30 with S32, we found that the precisions when categorical information not used were higher than the precisions when it was used. So, at least for these data, using category information was not a good thing.
- The automatic feedback method was always effective.
- The down-weighting method sometimes produced better results and sometimes produced poorer results.

2.8 Summary

System A uses such characteristics of newspapers as locational information and obtained good results in the CC Tasks. By performing comparative experiments, we confirmed that locational information was effective. The other kinds of information were, however, not so effective.

System A has many parameters and many methods. In the future, we would like to conduct much more extensive experiments in order to examine the effects of parameters and methods in System A.

3 Japanese and English IR Tasks

3.1 Overview of the results

The average precisions for System-B against relevant documents on JJ, EJ, EE, and JE tasks are presented in Table 2. In Table 2, 'very short' means that the system used the 'TITLE' part of the queries for retrieval, 'short' means that it used the 'DESCRIPTION' part of the queries, and 'long' means that it used all parts of the queries except the 'FIELD' part. For each task, 'feedback' means the precisions that were obtained by automatic-feedback retrieval, while 'initial' means the precisions that were obtained by using the raw initial queries. The symbol '$*$' means that the corresponding search results from System-B were submitted to the NTCIR 2 workshop committee as formal runs.[11] For the JJ and EE tasks, only 'feedback' results from System-B were submitted, while for the EJ and JE tasks, both 'initial' and 'feedback' results were submitted. These average precisions place the system in the highest-scoring group among those for which results were submitted.

We describe System-B in detail below. We start by describing the scoring function used to rank documents. Next, we describe the design issues involved in selecting possible free parameters and then compare results for various parameter values through experimented results. Finally, we conclude this section with a brief summary.

3.2 Scoring function

Our scoring function is based on BM11 [9]. Let D be a document and Q be a query, where D and Q have been tokenized into words. D and Q are bags of words. We define $|X|$ as the number of words in X and define $tf(w|X)$ as the number of a word w in X. We also define $W(X)$ as the set of different words in X.

The score of D given Q, $score(D|Q)$, is defined as:

$$score(D|Q) = \sum_{w \in W(D) \cap W(Q)} d(w|D) q(w|Q), \quad (9)$$

where $d(w|D)$ is the weight of w given D and $q(w|Q)$ is the weight of w given Q. $d(w|D)$ is defined as:

$$d(w|D) = \frac{tf(w|D)}{tf(w|D) + |D|/\Delta}, \quad (10)$$

where Δ is the average of $|D|$ over the document collection $\mathcal{D}$ that contains D,i.e.,

$$\Delta = \sum_{D \in \mathcal{D}} |D| / |\mathcal{D}|, \quad (11)$$

where $|\mathcal{D}|$ is the number of documents in $\mathcal{D}$. $q(w|Q)$ is defined as:

$$q(w|Q) = \frac{(k_q + 1) tf(w|Q)}{k_q + tf(w|Q)} idf(w), \quad (12)$$

where $k_q = 1000$ and

$$idf(w) = \log \frac{|\mathcal{D}|}{|\mathcal{D}(w)|}, \quad (13)$$

where $|\mathcal{D}(w)|$ is the number of documents that contain w. $\mathcal{D}(w)$ is , of course, a subset of $\mathcal{D}$.

$score(D|Q)$ is used for the initial search. For an automatic feedback search, we use $Score(D|Q)$:

$$Score(D|Q) = \sum_{w \in W(D) \cap W(Q')} d(w|D) q'(w|Q), \quad (14)$$

where

$$q'(w|Q) = \alpha q(w|Q) + \frac{\sum_{i=1}^{R} q(w|F(D_i))}{R}, \quad (15)$$

[11] On JJ short, System-A outperformed System-B. Its best average precision was 0.3730

Table 2. Average Precision (Relevant).

		very short	short	long
JJ	initial	0.2112	0.3082	0.3807
	feedback	0.2706*	0.3396*	0.4303*
EJ	initial		0.2497*	0.3156*
	feedback		0.2564*	0.3260*
EE	initial	0.2192	0.2714	0.3684
	feedback	0.2523*	0.3131*	0.4043*
JE	initial		0.3409*	0.3855*
	feedback		0.3413*	0.3856*

'*' represents submitted runs.

where α is a number , D_i is the top i-th document retrieved by initial search, R is the number of top-scoring documents used in the automatic-feedback search, and F is the function used to select appropriate terms from a document. Q' in Equation (14) is defined as:

$$Q' = Q \cup F(D_1) \cup \cdots \cup F(D_R). \qquad (16)$$

3.3 Design Issues

The free parameters we consider in this paper are α, F, and R in Equation (15). We tried to have these parameters defined automatically. Before, however, we describe our attempts at determining these parameters, we will discuss how we preprocessed documents and queries for the JJ, EE, JE, and EJ tasks.[12]

3.3.1 Tokenization

Tokenization is, to a large degree, language dependent.

We tokenized Japanese texts (documents or queries) by using ChaSen version 2.02[13] [4] and then extracted lemmas of content words as D or Q. We postprocessed the output of ChaSen to eliminate some erroneous patterns of tokenization.

In a similar way, we used LimaTK[14] to morphologically analyze English texts and then used a stemmer that built around a library available in the WordNet1.6 package[15] to lemmatize content words. Stop words were removed according to the list in the Nice stemmer package.[16]

The documents and queries thus processed were used for the JJ and EE tasks.

3.3.2 Query translation

For the JE and EJ tasks, we translated queries. Once we translate queries, cross-lingual IR (CLIR, i.e., JE or EJ) is performed by the same method as used for mono-lingual IR (JJ or EE). We describe the method below as applied to the translation of a Japanese query into English. English to Japanese translation is performed in a similar way.

We perform document expansion [15] to augment the original queries; i.e., for a Japanese query, we first search the Japanese database to get documents that are relevant to the query. Next, we extract the words contained in the top-5 documents and combine them to the original query. We thus obtain an expanded Japanese query.[17]

The expanded Japanese query is then translated into English. For the translation, we first made a Japanese-to-English bilingual dictionary from the Japanese-English abstract pairs provided for the first NTCIR Workshop. From those pairs, we extracted Japanese-English keyword pairs contained in the abstract pairs. It was possible for these keywords to be phrases or words. If a Japanese keyword co-occurred with multiple English keywords, then we selected the most frequently co-occurring English keyword as the translation of the Japanese keyword [2]. Texts were translated in the following two steps; we used ChaSen to morphologically analyze the text, then translated the sequence of morphemes into English. The translation was on a word-to-word or phrase-to-phrase basis. Disambiguation by contexts was not used. The translation was based on longest matches. For example, if a query 'a b c' is given, where 'a' is translated into 'A' and 'a b c' is translated into 'D E', then 'a b c' is translated into 'D E'.[18]

[12] The method used to preprocess documents and queries for CC tasks is similar to, but more primitive than, a method described in section 2. We, thus, omit a description here.

[13] `http://chasen.aist-nara.ac.jp/`

[14] `http://cl.aist-nara.ac.jp/~tatuo-y/ma/`

[15] `http://www.cogsci.princeton.edu/~wn/`

[16] `http://www.ils.unc.edu/iris/irisnstem.htm`

[17] Local context analysis has been used to expand queries in CLIR [1]. The comparison is a future work.

[18] [2] also used a longest-match algorithm, but they did not use a morphological analyzer, which might degrade the system performance. This belief is supported by Table 3 which shows the performance of our method in no document expansion. The average precision of [2] on the same task was 0.3216, while that of our approach is 0.3364.

Translated queries were used for the JE and EJ tasks. The retrieval algorithm was the same as that used for the JJ and EE tasks.

As is shown in Table 2, our approach to the JE and EJ tasks worked quite well. It is evident, however, that the degree of success of our approach depends on the degree of similarity between the Japanese database and the English database used for CLIR. We thus conducted another experiment which used the databases and JE-queries provided for the first NTCIR Workshop. The type of query used for the experiment was 'long' except that we did not use English concepts.

Table 3. Average precisions with document expansion.

Source	Target	Average precision
ϕ	ntc1-e	0.3364
ntc2-j	ntc1-e	0.3628
ntc1-j	ntc1-e	0.3899

In Table 3, the column 'Source' lists the databases used to expand the original queries. 'ϕ' indicates no document expansion. 'ntc2-j' means that the Japanese database which was freshly added for the second NTCIR Workshop was used for document expansion, and 'ntc1-j' means that the Japanese database provided for the NTCIR workshop 1 was used for document expansion. 'ntc1-e', which is listed in 'Target' column for all entries, is the English database that was the target of the searches for documents. Average precision was evaluated against relevant documents in 'ntc1-e'.

'ntc1-j' and 'ntc1-e' are nearly parallel. Naturally, it achieved the best performance of these three cases. 'ntc2-j' and 'ntc1-e' are comparable. The average precision is still better than with no document expansion. Document expansion is thus worthwhile for CLIR.

We have briefly described the language-dependent parts of System-B. Next, we describe its language-independent parts, describing F, R, and α in Equation (15), in that order.

3.3.3 Definition of F

We define a relevance of word w for the top-scoring R documents in terms of probability.[19]

Given a bag of words X, then the probability of w, $\Pr(w|X)$, and its variance $\mathrm{Var}(w|X)$ are estimated as

$$\Pr(w|X) = \frac{tf(w|X) + 1}{|X| + 2}, \tag{17}$$

$$\mathrm{Var}(w|X) = \frac{\Pr(w|X) * (1 - \Pr(w|X))}{|X| + 3}. \tag{18}$$

We then define D_R^1 as the bag of words that contains all the words in $D_1, D_2, \cdots, D_R$ and define $\overline{D_R^1}$ as the complement of D_R^1 with a universal set that is defined by all of the words in the document collection $\mathcal{D}$.

The relevance of word w, $rel(w|D_R^1)$, is defined as

$$rel(w|D_R^1) = \frac{\Pr(w|D_R^1) - \Pr(w|\overline{D_R^1})}{\sqrt{\mathrm{Var}(w|D_R^1) + \mathrm{Var}(w|\overline{D_R^1})}}. \tag{19}$$

Finally we define $F(D_i)$ as

$$F(D_i) = \{w | rel(w|D_R^1) >= \theta \wedge w \in D_i\}, \tag{20}$$

where θ is a predefined threshold.

$rel(w|D_R^1)$ approximately follows the standard normal distribution. Possible candidates for θ are 1.28, 1.65 and 2.33, which correspond to significance levels of 0.10, 0.05 and 0.01, respectively. Hereafter, significance levels are represented by p.

We used $p = 0.10 (\theta = 1.28)$ for all of the submitted runs.[20] This choice was based on previous experiments conducted on the database provided for the first NTCIR Workshop. $p = 0.10$ is a robust parameter value for term selection as is shown in section 3.4.

3.3.4 Definition of R

We used the method explained below to set R automatically. We found, however, that the method was not efficient, and this is shown in section 3.4.

Our method is based on the degree of increase in the number of different words in top-scoring documents. If the content of successive documents is similar, the documents should share keywords. This degree of increase is thus low when similar documents continue. Our algorithm is depicted in Figure 2. In the experiments described in section 3.4, the average numbers of documents selected by the algorithm were 3.81, 4.01, and 3.95, for 'very short', 'short', and 'long' queries, respectively.

As is shown in section 3.4, the performance of IR is quite sensitive to R. We will therefore investigate methods for the automatic-determination of R in future work, though our initial attempts have not been too successful.

3.3.5 Definition of α

α is defined heuristically as follows:

$$\alpha = |W(F(D_R^1))|^{\frac{1}{|W(Q)|}}, \tag{21}$$

where

$$F(D_R^1) = F(D_1) \cup F(D_2) \cup \cdots \cup F(D_R). \tag{22}$$

$\alpha \geq 1$ holds because $|W(F(D_R^1))| \geq 1$. α approaches 1 when $|W(Q)|$ is large. α takes a large value when the number of different words in Q is small and the

[19] [10] also uses a probabilistic metric to select relevant terms.

[20] We used a more precise value for θ actually.

```
for(R=3;;R++){
  if (diff(R) > diff(R-1)) {
    break
  }
}

int diff(i) {
   return |W(F(D_i^1))| − |W(F(D_{i−1}^1))|
}
```

Figure 2. Algorithm for determining R.

number of different words in D_R^1 is large. α is defined so that Q is more important than D_R^1. In the experiments described in section 3.4, the average value of α were 13.44, 3.89, and 1.14, for 'very short', 'short', and 'long' queries, respectively.

This heuristic approach worked reasonably well as is shown in section 3.4.

In summary, for the formal runs, we used $p = 0.10$ for term selection and used automatic methods to set R and α. $p = 0.10$ was the only parameter that we had to set by hand.

3.4 Comparison of parameter values

We varied the values of p, R, and α to observe the effects of parameter values on performance. Performance was measured by the average precision against relevant documents. We used the queries and documents provided for the second NTCIR workshop. Experiments were conducted on JJ and EE tasks. We only report on the results for JJ tasks, here, because both sets of results displayed the same tendency.

The parameter values for p were

$$p = 0.10, 0.05, 0.01. \tag{23}$$

The parameter values for R were

$$R = 1, 3, 5, 7, 10, 15. \tag{24}$$

The parameter values for α were

$$\alpha = 0.5, 1.0, 1.5. \tag{25}$$

For R and α, we also tried the heuristic methods described in Figure 2 and Equation (21). We tried all combinations of these parameter values. Thus, we conducted $3 \times 7 \times 4 = 84$ runs to make our comparison for each of the 'long', 'short', and 'very short' queries.

To evaluate the effectiveness of a parameter, we fixed its value and then calculated the average of the average precisions of the 84 runs. The results are shown in Figures 3, 4, and 5. In these figures, horizontal axes represent the query types and vertical axes represent the average precisions. The title of each line indicates the parameter value. 'var' means that values are determinbe by our methods proposed above. 'initial' means the results for the initial search. The titles are in order of decreasing average precision for short queries.

Figure 3 shows the results for various settings of p. Note that $p = 0.1$ and $p = 0.05$ performed equally well. This suggests that the value of p is robust over this range.[21]

Figure 4 shows the results for various settings of R. It is difficult to detect any clear tendency in Figure 4, but it seems that when queries are long, small R values perform well, and when queries are short, large R performs well. This suggests that the length of queries could be used to set R automatically.

Figure 5 shows the results for various settings of α. The average of α were 13.44, 3.89, and 1.14 for 'very short', 'short', and 'long' queries, respectively. α takes large values for 'very short' and 'short' queries. It takes small values for 'long' queries. α worked reasonably well. This is because for 'very short' and 'short' queries, the results of the initial search are not very reliable, so we had to weight Q heavily, while for 'long' queries, the results of the initial search are reliable, so we don't have to weight Q so heavily.

3.5 Summary

System-B was designed as a portable IR system that avoids free parameters as much as possible. It will be possible to improve the sysmtem's performance by providing a proper method for determining the number of top-ranked documents to be used in automatic-feedback.

4 Conclusion

We have developed two systems for the second NTCIR Workshop IR tasks. One was an improved version of the system that was used for the first NTCIR Workshop IR tasks and the IREX Workshop IR tasks. The other was a freshly developed system that avoids free parameters as much as possible. The former system participated in the JJ and CC tasks and the latter system participated in the JJ, EE, JE, EJ and CC tasks. Both systems achieved good results. We have not yet compared the two systems thoroughly. In the future, we will conduct a more detailed examination of our systems and will determine what kinds of information are effective.

[21] Additional experiments showed that average precisions for $p = 0.9$ to 0.05 performs equally well. (The results of $p = 0.1$ were slightly better than those of other values.)

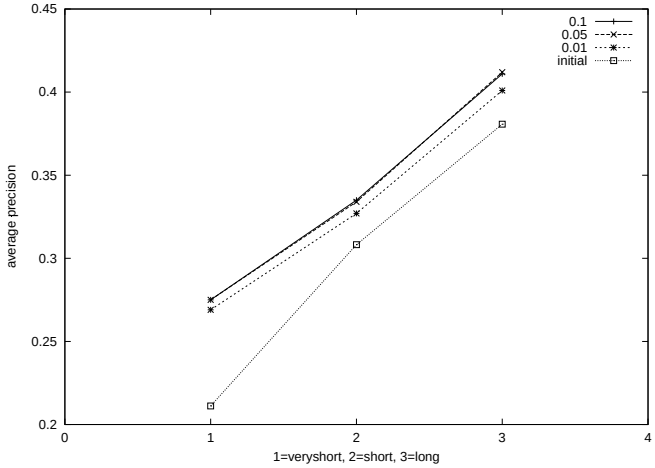

Figure 3. Average precisions for various settings of p

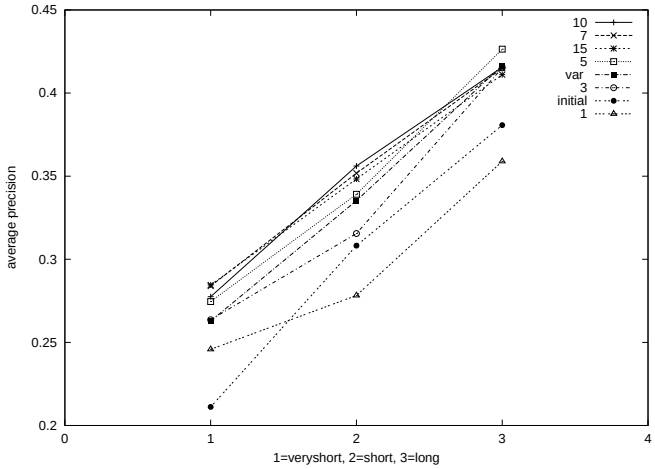

Figure 4. Average precisions for various settings of R

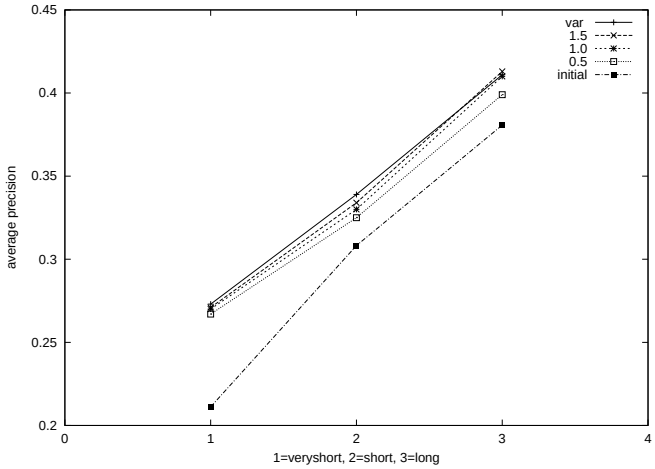

Figure 5. Average precisions for various settings of α

References

[1] L. Ballesteros and W. B. Croft. Phrasal translation and query expansion techniques for cross-language information. In *Proc. of SIGIR'97*, pages 84–91, 1997.

[2] A. Chen, F. C. Gey, K. Kishida, H. Jiang, and Q. Liang. Comparing multiple methods for Japanese and Japanese English text retrieval. In *Proc. of NTCIR Workshop 1*, pages 49–58, 1999.

[3] S. Fujita. Notes on phrasal indexing JSCB evaluation experiments at IREX-IR. *Proceedings of the IREX Workshop*, pages 45–51, 1999.

[4] Y. Matsumoto, A. Kitauchi, T. Yamashita, and Y. Hirano. Japanese morphological analysis system ChaSen manual. Nara Institute of Science and Technology, 1999.

[5] M. Murata, K. Uchimoto, H. Ozaku, and H. Isahara. Information retrieval based on stochastic models. In *Proc. of NTCIR Workshop 1*, pages 59–70, 1999.

[6] M. Murata, K. Uchimoto, H. Ozaku, and Q. Ma. Information retrieval based on stochastic models in IREX. In *Proc. of the IREX Workshop*, pages 33–40, 1999.

[7] M. Murata, K. Uchimoto, H. Ozaku, Q. Ma, M. Utiyama, and H. Isahara. Japanese probabilistic information retrieval using location and category information. *The Fifth International Workshop on Information Retrieval with Asian Languages*, 2000.

[8] T. Ozawa, M. Yamamoto, H. Yamamoto, and K. Umemuru. Word detection using the similarity measurement in information retrieval. *Proc. of the 5th Conference on Applied Natural Language Processing*, pages 305–308, 1999. (in Japanese).

[9] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proc. of SIGIR'92*, pages 232–241, 1992.

[10] S. E. Robertson and S. Walker. Okapi/Keenbow at TREC-8. In *Proc. of TREC'8*, 1999.

[11] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *TREC-3*, 1994.

[12] J. J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System*, pages 313–323. Prentice Hall, Inc., 1971.

[13] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. In K. S. Jones and P. Willett, editors, *Readings in Information Retreival*. Morgan Kaufmann Publishers, 1997.

[14] S. Sekine and H. Isahara. IREX project overview. *Proceedings of the IREX Workshop*, pages 7–12, 1999.

[15] A. Singhal and F. Pereira. Document expansion for speech retrieval. In *Proc. of SIGIR'99*, pages 34–41, 1999.

[16] R. Sproat and S. Shih. A statistical method for finding word boundaries in Chinese text. In *Computer Processing of Chinese & Oriental Languages*, pages 336–351, 1990.

[17] M. Sun, D. Shen, and C. Huang. CSeg&Tag1.0: A Practical Word Segmenter and POS Tagger for Chinese Texts. In *The Fifth Conference on Applied Natural Language Processing, Association for Computational Linguistics*, pages 119–126, 1997.